# CANMET

Canada Centre for Mineral and Energy Technology

Centre canadien de la technologie des minéraux et de l'énergie

## Energy Research Laboratories

## Laboratoires de recherche sur l'énergie

Energy, Mines and Resources Canada

Énergie, Mines et Ressources Canada

Canada

THE ENERGY OF OUR RESOURCES • THE POWER OF OUR IDEAS

L'ENERGIE DE NOS RESSOURCES • NOTRE FORCE CREATRICE

**Development of PC Control - Phase III**
Final Report

Jagdish S. Patel
Atlantis Scientific Systems Group Inc.

Work on the project was conducted under the auspices of
the Canadian Centre for Mineral and Energy Technology,
Energy, Mines and Resources Canada.
This work was supported by the Federal Panel on Energy R&D (PERD)

DSS FILE NO: 33SQ.23440-1-9004

DSS CONTRACT SERIAL NO: ~~23340~~ 23440-1-9004/01-SQ

CANMET PROJECT NO: 501202F1

CANMET SCIENTIFIC AUTHORITY: D. J. PATMORE

DATE: November 1, 1991

13 pp

MICROMEDIA

92-06751/

**Executive Summary of the Development of PC Control - Phase III**
Contract No. 23440-1-9004

Jagdish S. Patel
Atlantis Scientific Systems Group Inc.
1827 Woodward Drive, Ottawa, Ontario, K2C 0P9, Canada.

November, 1991

This contract involved the continuing development of a process monitoring and control software package, PC Control. The developments comprised the following. An enhanced device driver for the Micromac instrument to interpret overload measurements and a display mechanism for overload measurements. A capability to send data reports to the printer, or to a disk file, or to both, with improved error handling. A powerful online historical trending facility, combined with a new data logging system.

**Résumé du Développement de PC Control - Phase III**
N$^{O}$ de contrat 23440-1-9004

Jagdish S. Patel
Atlantis Scientific Systems Group Inc.
1827 Woodward Drive, Ottawa, Ontario, K2C 0P9, Canada.

Novembre, 1991

Le présent contrat porte sur le développement continu d'un progiciel de contrôle et de surveillance des processus, appelé PC Control. Les projets de développement concernent ce qui suit: vue d'interpréter les mesures de surcharge; un mécanisme afficheur des mesures de surcharge; une fonctionnalité possédant une manipulation améliorée des erreurs pour l'envoi des rapports de données à l'imprimante ou à un fichier sur disque; un module performant en ligne des tendances historiques couplé à un nouveau système de registre.

**Final Report**
**Development of PC Control - Phase III**
Contract No. 23440-1-9004

by
Jagdish S. Patel

Atlantis Scientific Systems Group Inc.
1827 Woodward Drive, Ottawa, Ontario, K2C 0P9, Canada.

November, 1991

1. **Introduction** . . . . . 5

2. **Task 1: Modification of the Driver Program for the uMAC** . . . . . 5
   - 2.1 Schematic Display of Overload/Error Tag Values . . . . . 6
   - 2.2 Reporting of Overload/Error Tag Values . . . . . 6
   - 2.3 "Show" Description of Overload/Error Tag Values . . . . . 6

3. **Task 2: Modification of the Targets for Generation of Reports** . . . . . 6

4. **Task 3: Online Historical Trending** . . . . . 7
   - 4.1 Binary Data Logging System . . . . . 8
   - 4.2 Data Retrieval System . . . . . 9
   - 4.3 LISTBIN - Program to List Binary Data Files . . . . . 12
   - 4.4 Programming Example to Read Binary Data Files . . . . . 12

5. **Other Additions, Notes and Corrections** . . . . . 13

## 1. Introduction

This document describes software design and implementation of new features to the PC Control software, as related to contract 23440-1-9004. The accompanying manual, "**Users Manual - PC Control Development, Phase III**" describes, from a users point of view, the new features added by this contract. The user is advised to read both documents to understand more fully the design intent.

PC Control is a software system that allows powerful real time monitoring and control of the CANMET Hydrocracking units, based on IBM PC compatible computer hardware.

Three major tasks, as outlined in Appendix "A" of the contract, were executed in this contract. All changes affected only the foreground program of PC Control. The new version of the foreground program works with the previous background program, "CONTROL". In the following sections, the design and development of each of the tasks is described.

## 2. Task 1: Modification of the Driver Program for the uMAC

The Micromac, abbreviated uMAC, is one of the main measurement and control devices attached to the PC Control computer. The uMAC has its own microcomputer, and conveys measured data to the PC Control computer via a serial communications line. For level input (LI) points, the measured data obtained by the PC Control computer is in the form of a string of characters, usually representing the numeric data values.

Occasionally, the string of characters contains the word "OVLOAD", in place of the numerical value, which means that the analog to digital converter on board the uMAC received a measurement signal value beyond its operating range.

In previous versions of PC Control, the overload messages were not decoded, and the tag value was kept the same as the previous tag value. Hence, the user was unable to differentiate a valid value from an assumed value, because of the overload situation.

The new version of the software has an enhanced uMAC driver, which is able to detect the overload situation, as well as other errors from garbled characters. Since the actual value of a point is not known in cases of an overload or other errors, the **previous value of a tag is used for the current value.**

However, the user is informed of an overload or other errors by one of three ways:

1) In Display of the process schematic

2) In Reporting: Printed report of tag values

3) "Show" menu item: Description of a tag's configuration.

The following describes each of the ways.

## 2.1 Schematic Display of Overload/Error Tag Values

If a measured value (currently, any LI tag associated with the uMAC device) is in error or has overloaded the measuring device, then the display of that point alternates between "???" and the assumed previous value of the tag.

This form of display is an indicator to the operator of an overload or error situation has occurred. Since an operator may not always be present at the terminal, these errors are **latched** so that the display continues to indicate the error situation, even if it has passed.

Once the operator has seen the latched errors, the display can be **cleared** by pressing **any key**.

## 2.2 Reporting of Overload/Error Tag Values

The presentation of tag values in reports are also noted by "???" printed after the data values, if the tag values are overloaded or in error, **at the time of printing the report**. Of course the printed tag value is the previous value if an overload or error occurred.

Note that only **current**, ongoing overload or error situations are shown as "???" in the reports.

## 2.3 "Show" Description of Overload/Error Tag Values

When you use the **Show** menu item of the Tags Editor, a summary of the current tag's configuration is shown. For LI tags, overloads and errors are indicated next to the tag value by "???" and by the words "**Overload**" or "**Error**".

Note that only **current**, ongoing overload or error situations are shown as "???" by the Show menu item.

# 3. Task 2: Modification of the Targets for Generation of Reports

The new version of PC Control allows report generation to

1) a disk file,
2) to the printer or
3) to both a disk file and the printer.

The third choice is provided in the new version, and it allows backup reports to be generated to the disk drive even if the printer experiences a fault, such as running out of paper.

In addition to providing the choice of both devices for reporting, the reporting function itself has been re-written to be more robust in case of device failure (Hard disk or Printer), as explained below. If any one of these devices has a problem, you will see one of the following error messages:

error messages:

DISK ERROR: Cannot write Report C:\PCC\MMHHDDMM.RPT

or

PRINTER ERROR: Cannot print Report.

If a printer is experiencing a fault, up to two minutes may pass before the error message is shown. PC Control continues to write the report to any designated device as long as it is functioning. Hence if reporting was active to both devices, and the printer experiences a paper jam, then the report is not printed there until the printer is fixed, however the reporting to disk is continued, as long as it too experiences no faults. Once any device fault is corrected, PC Control will automatically start printing reports to that device.

The selected target device for report selection is saved in the environment file, PCC36.ENV, and becomes the default until changed.

This feature was designed, tested and delivered in an advanced release of the software.

## 4. Task 3: Online Historical Trending

This feature, in conjunction with the binary data logging system, allows the user to do true historical trending. The user is allowed to pick any valid time interval. A valid time interval is that which lies within the total time span covered by the set of existing binary data files. The binary data files are logged by the logging function of the PC Control program. Additionally, the user can select up to six tags from any of the available tags to be viewed simultaneously.

There are two components to this feature:

1) A **Binary Data Logging System** in place of the Lotus-compatible files, described in Section 11.6 of the PC Control User's Manual, and

2) An intelligent **Data Retrieval System**, with a generalised design to support multiple output formats, and currently allows the retrieved data to be passed to the graphing function.

The next two sections describe these components in detail. In other sections, a programming example is provided to users wishing to read the binary files into their own programs. Finally, a utility program, **LISTBIN**, which is supplied with the final release of the PC Control software is described, which allows the user to list the time ranges of binary files.

### 4.1 Binary Data Logging System

The binary data logging system supplants the previous logging system, which produced Lotus-123 readable files. The new logged files are binary and not "ASCII text", which means that you cannot view them easily with a text editor or a word processor. They also cannot be read

you cannot view them easily with a text editor or a word processor. They also cannot be read into a Lotus-123 compatible spreadsheet program without format conversion. Instead, the tag values are stored in a compact, native binary format, the same as used internally in PC Control.

The binary format is chosen because it offers several advantages over the previous Lotus-compatible "number import" file format:

1) The files are very fast to write and read

2) Binary data files are much smaller in size

3) The new file system writes the data of all 100 tags, rather than a selection of tags

4) Binary data files store the full precision of the tag data, not a rounded-off value

5) Each row of tag data is time-stamped with the actual system time

The speed of reading and writing the new log files comes about from its regular format which can be thought of as rows of data; each row occupies exactly 404 bytes, containing two components as described:

- **Date and time of the sample.** It is more precisely, the number of seconds elapsed since 00 hours 00 minutes 00 seconds on Jan 1, 1976, which in turn is based on the Julian Day calendar. This field is of computer type "long integer", occupying 4 bytes.

- **Tag values of tag 0 to tag 99.** The tag values are of computer type "single precision floating point number", occupying 4 bytes for each value. The tag values are packed one after another, without space characters. The level input and level output tags are represented by 0.0 if a given tag is OFF and 1.0 if the tag is ON. The total size of this component is 400 bytes.

PC Control's internal representation of the tag values is also of computer type "single precision floating point number, which is why the full precision of the number can be saved to a file.

The small size of the binary files comes about because a binary number converted to a printable decimal number takes more space than the original binary number. It has been shown that the binary files can further be "compressed" by programs such as PKZIP to about 25% of its original size, resulting in even more savings in storage. This can be important when you wish to archive the logged data files.

### 4.1.1 File naming convention

The files generated by the new binary data logging system are named in a similar way to the previous Lotus compatible files. The file extension is **".BDA"**, to differentiate these files from

the Lotus compatible files, which has an extension ".DAT". The data files are kept in the directory **"\PCC\DATALOG"**, as before.

As before, the filename is made up of the month, day, hour and minute of the time of logging the file, **"MMDDHHMM"**.

The DOS date and time stamp of the file corresponds exactly to the latest or last sample saved to the file. All other data are samples older than the last, each separated by the sample period. It is important not to modify this date and time stamp, since the Data Retrieval System uses this information.

## 4.2 Data Retrieval System

The data retrieval system was designed as the complementary tool for extracting historic data of any period. The extracted data is displayed in a graphical format. It has been carefully designed for high speed; It obtains the required data from the logged files very quickly.

The system is "intelligent" in that the user is only allowed to enter a time period that is valid; the data must actually exist. Also, this time period is remembered as a default for the next time you use the Data Retrieval System. Furthermore, the user is able to enter the time period either by editing the begin and end times of the desired plot, or by selecting specific data files, or both. The editor associated with the selection of the begin and end times of the plot also displays the day of the week as a guidance to the user.

Access to the is by selecting these menu items from the root menu: **Graph -> Historical.** Once in the Data Retrieval System, the user is shown a screen with two panels, as depicted in Fig. 1:

- o An upper **activity window**. This scrolling window shows detailed activity of the Data Retrieval System, and in general, the user need not worry about everything shown here, unless if an error message or a prompt is shown.

- o A lower **edit window**. This fixed window shows two rows of date and times, the upper being the begin time and the lower the end time of the plot.

Below the edit window, a help line is shown that summarises the functions available. This line may be cleared and replaced by the usual system informational or error messages since PC Control continues its monitoring and control functions. The user may recall the help line again by pressing the F1 key.

When you begin a session, the Data Retrieval System quickly determines and then displays the date and times of the very first and last data points available in the available set of data files. The total number of files present is also shown like this:

```
Files found      : 102
Interval available:
```

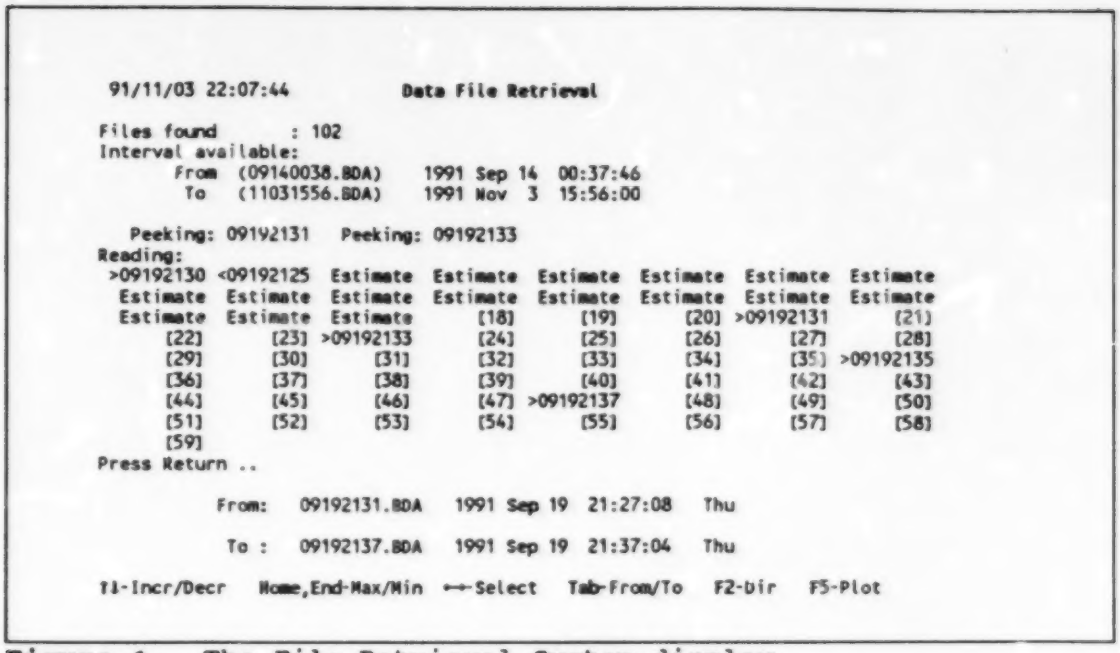


**Figure 1.** The File Retrieval System display

From (09140038.BDA) 1991 Sep 14 00:37:46
To (11031556.BDA) 1991 Nov 3 15:56:00

A maximum of 500 data files will be examined, or until the free computer memory runs out, whichever happens first. If this occurs, the user will see the following error message:

**"(Cannot read more files)".**

This is only a warning message, since the system will still work, even with a subset of all the possible data files. The user should remember this fact though, and attempt to move unnecessary binary data files out of the \PCC\DATALOG directory. It is a good idea to archive these data files to removable disks from time to time anyway.

The previously chosen begin and end times are shown by default, if these times are still valid. The editor is simple to use, and is similar to other PC Control editors.

The user will not be able to set the begin time in advance of the end time. By default, you are allowed to edit the time values. But if you move the cursor left over to the file names, then the file name updates to the nearest file that contains the data.

The file name field can be "incremented" and "decremented" by one just like any of the time fields. In this case, the corresponding date and time is updated automatically. This feature is very useful in letting you pick the exact begin and end times corresponding to one or more files.

Once the begin and end times have been edited to your needs, you may proceed by pressing the F5 function key to plot that data.

### 3.2.1 Data Retrieval Method

Once the F5 function key is pressed, all the relevant file(s) are read to obtain 60 points that lie in the specified interval. Normally, for a given time, its associated data exist in some file or other; The system attempts to pick that data whose time matches as closely as possible to the time in question, without doing any interpolation. The process of re-sampling the data has all the usual phenomena that can be expected from this method, and the user should be aware of this. For example, aliasing can arise when rapidly fluctuating tag values are not sampled at a high enough rate. So, if you plot data that was obtained with one time base on another larger time base, then all the rapidly fluctuating signal detail is not shown correctly.

A situation can arise when for a given time, no corresponding data can be found in any of the files. However, adjacent data files do exist. In this case, where some data files are missing, the Data Retrieval System does a (linear) interpolation between the nearest known data points whose time lies about the time in question. When this happens, the activity window prints the message **"Estimate"**, telling the user that an interpolation is done. The resulting plot may not be a correct representation of the actual history.

After all the data is read, the user must press a carriage return to commence plotting. After viewing the plot, the user may exit after pressing the escape key, and is returned to the Data Retrieval System.

### 4.2.2 Choosing Tags to Plot

Up to six tags can be graphed simultaneously using the Data Retrieval System. The tags used are the same as those used for the Trend6 graphing function. Hence, you use the same procedure for selecting these as before, and this include the use of the Graph Settings Save and Recall Capability, as described in the Addendum to PC Control Manual.

### 4.3 LISTBIN - Program to List Binary Data Files

This useful program is a stand alone utility was designed initially for testing purposes, during design of the Data Retrieval System. It has been included in the delivery software package. LISTBIN lets you look at the time interval of each of the binary data files in the **"\PCC\DATALOG"** directory. To invoke this program, make certain that you are logged to the correct disk drive, then type LISTBIN at the DOS prompt. A chronologically sorted list of the files is presented, as shown by the example output in Fig. 2.

### 4.4 Programming Example to Read Binary Data Files

```
11031512.BDA 60 499792206 499792324 ¦ 1991/11/ 3 15:10: 6 ¦
1991/11/ 3 15:12: 4
11031514.BDA 60 499792322 499792440 ¦ 1991/11/ 3 15:12: 2 ¦
1991/11/ 3 15:14: 0
11031531.BDA 46 499793370 499793460 ¦ 1991/11/ 3 15:29:30 ¦
1991/11/ 3 15:31: 0
11031533.BDA 60 499793462 499793580 ¦ 1991/11/ 3 15:31: 2 ¦
1991/11/ 3 15:33: 0
11031538.BDA 60 499793762 499793880 ¦ 1991/11/ 3 15:36: 2 ¦
1991/11/ 3 15:38: 0
11031540.BDA 60 499793882 499794000 ¦ 1991/11/ 3 15:38: 2 ¦
1991/11/ 3 15:40: 0
11031550.BDA 17 499794568 499794600 ¦ 1991/11/ 3 15:49:28 ¦
1991/11/ 3 15:50: 0
11031554.BDA 20 499794802 499794840 ¦ 1991/11/ 3 15:53:22 ¦
1991/11/ 3 15:54: 0
11031556.BDA 60 499794842 499794960 ¦ 1991/11/ 3 15:54: 2 ¦
1991/11/ 3 15:56: 0
```

**Figure 2.** Example output from LISTBIN. The output presents the file name, number of data rows of data, the begin time in seconds, the end time in seconds (elapsed since Jan 1, 1976), the begin time in date/time and the end time in date/time.

An example procedure, or subroutine that reads a binary data file and prints out some tag values is shown in Fig. 3. This example is written in the Pascal programming language, but may be "translated" to the user's choice of programming language. To use this procedure, one simply makes a call to the subroutine in the main part of the program, as follows:

```
program ReadBinaryDataFile;
begin
  ...
  ...
  ReadBinDataFile('11031512.BDA');
  ...
  ...
end.
```

Note that the procedure is reading all the rows of a file, that is, until the end-of-file function **eof** becomes true. For each row, the procedure prints the values of tag numbers 16 to 22 inclusive on a separate line.

## 5. Other Additions, Notes and Corrections

1. (For ERL version, CANMET building 4). The previous version of the background was hardcoded to exclude Tag 66:WT91 from alarm logs, because the alarms were used for control purposes. The new version excludes instead, Tag 87:HCPLVL, which is used for the equivalent control.

2. (For ERL version, CANMET building 4). The previous diagnostic program for the Micromac sub-system has been updated to show interpretation of overload and other errors. For more information, see Section 2 of this article.

Several other small improvements were also added to the foreground program of PC Control, that has not changed the functionality of the program.

```
{++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++
++++++++++++++
The following procedure may be called from the main program to
read a
binary data file.  The input parameter is the file's name, eg
'ABC.BDA'
}

type
  String20 = string[20];
  BinData_t = record                          { a row of data has:
}
              T: longint;                          { time - # of
seconds }
              D: array[0..99] of Single;   { 100 tag values, in
single precision }
            end;

procedure ReadBinDataFile(FN: String20);
var
  bd1:          BinData_t;
  Fbd1:         file of BinData_t;
  row_num:      integer;
  tag_num:      integer;

begin    {ReadBinDataFile}
    Assign( Fbd1, 'C:\PCC\DATALOG\' + FN );
    Reset( Fbd1 );
    if (IOResult = 0) then                { check there are no
disk errors }
    begin
      writeln;                             { print line feed }
      writeln('Reading file: ', FN);
      row_num := 0;                       { number of rows of data
read }
      while not eof( Fbd1 ) do
      begin
        Read( Fbd1, bd1 );                { read a row of data and
time stamp }
        write('Row :', row_num:3);          { formatted printing
like Row :NNN }
        row_num := row_num + 1;
        for tag_num := 16 to 22 do          { printing a small
selection here }
        begin
          write(bd1.D[tag_num]:5:1);
            { :5:1 formats to 1 place after decimal, and 5
characters total }
        end;
        writeln;                            { print line feed after
each row }
      end;
    end;
    Close( Fbd1 );
end;     {ReadBinDataFile}
```

**Figure 3.** A Pascal procedure (subroutine) to read a binary data file.